NASA TECHNICAL
MEMORANDUM

NASA TM X-64906

TECHNIQUES FOR TRAJECTORY OPTIMIZATION USING A HYBRID COMPUTER

By P. L. Neely
Systems Dynamics Laboratory

February 7, 1975

NASA

George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama

TECHNICAL REPORT STANDARD TITLE PAGE

1. REPORT NO. NASA TM X-64906	2. GOVERNMENT ACCESSION NO.	3. RECIPIENT'S CATALOG NO.

4. TITLE AND SUBTITLE Techniques for Trajectory Optimization Using a Hybrid Computer	5. REPORT DATE February 7, 1975
	6. PERFORMING ORGANIZATION CODE
7. AUTHOR(S) P. L. Neely	8. PERFORMING ORGANIZATION REPORT #
9. PERFORMING ORGANIZATION NAME AND ADDRESS George C. Marshall Space Flight Center Marshall Space Flight Center, Alabama 35812	10. WORK UNIT NO.
	11. CONTRACT OR GRANT NO.
12. SPONSORING AGENCY NAME AND ADDRESS National Aeronautics and Space Administration Washington, D.C. 20546	13. TYPE OF REPORT & PERIOD COVERED Technical Memorandum
	14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

Prepared by Systems Dynamics Laboratory, Science and Engineering

16. ABSTRACT

The use of a hybrid computer in the solution of trajectory optimization problems is described. The solution technique utilizes the Indirect Method and requires iterative computation of the initial condition vector of the co-state variables. Convergence of the iteration is assisted by Feedback Switching and Contour Modification. A simulation of the method in an on-line updating scheme is presented.

17. KEY WORDS	18. DISTRIBUTION STATEMENT Unclassified — Unlimited [illegible]

19. SECURITY CLASSIF. (of this report)	20. SECURITY CLASSIF. (of this page)	21. NO. OF PAGES	22. PRICE
Unclassified	Unclassified	47	NTIS

ACKNOWLEDGMENT

This work was performed for the Systems Dynamics Laboratory (formerly the Aero-Astrodynamics Laboratory) under a Postdoctoral Research Associateship sponsored by the National Academy of Sciences, National Research Council, Washington, D. C., with Dr. Judson Lovingood as scientific advisor.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

DEFINITION OF SYMBOLS

Symbol	Definition
$C \supseteq C_x, C_p$	Diagonal coefficient matrices determining feedback level
E	Criterion function for Contour Modification
$\underline{e} \supseteq \underline{e}_x, \underline{e}_p$	Vectors of boundary errors e_i, $i = 1, \ldots n$
$\underline{f}$	Vector of known functions f_k, $k = 1, \ldots n$
H	Matrix of second derivatives of Jm with respect to $\underline{p}(t_0)$
$\mathcal{H}$	Hamiltonian
J	Performance index
Jm	Augmented performance index
J_{max}	Upper bound on J
K	Terminal cost function
k	Step size
L	Integral cost function
m	Number of feedback stages
n	Number of state variables
$\underline{p}$	Vector of co-state variables p_i, $i = 1, \ldots n$
r	Number of control variables
t	Time
t_0	Initial time
t_f	Final time
$\underline{u}$	Vector of control variables u_j, $j = 1. \ldots r$

DEFINITION OF SYMBOLS (Concluded)

Symbol	Definition
$\underline{w} \supset \underline{w}_x, \underline{w}_p$	Vectors of feedback variables w_i, $i = 1, \ldots n$
$\underline{X}^0$	Present state
$\underline{X}^f$	Final state
$\underline{x}$	Vector of state variables x_i, $i = 1, \ldots n$
$\hat{\underline{x}}$	Augmented state vector x_i, $i = 0, 1, \ldots n$
$\underline{\alpha}$	Vector of weighting coefficients α_i, $i = 1, \ldots n$
ΔJ	Incremental change in J
Δp	Incremental change in $\underline{p}(t_0)$
$\Delta \tau$	Time to compute updated control
ϵ	Lower bound on $\Delta J/\Delta p$
μ	Fractional change in C
τ	Control updating interval

TECHNICAL MEMORANDUM X-64906

TECHNIQUES FOR TRAJECTORY OPTIMIZATION USING A HYBRID COMPUTER

INTRODUCTION

Numerous applications of optimal control theory have been made to the control and guidance of aircraft, launch vehicles, and spacecraft. In general, however, optimal guidance solutions have been evaluated off-line and any on-board guidance updating capability has been limited to quasi-optimal schemes such as the Iterative Guidance Mode [1]. The limiting factor in any on-board guidance updating scheme is the computing speed available. The solution of optimal control problems requires repeated integration of sets of differential equations — a time-consuming operation.

The proposed MASCOT system [2] would allow the on-board solution of optimal guidance problems by using new high-speed algorithms for numerical integration of the differential equations on a digital computer. An alternative approach is to use hybrid computation, with a high-speed model of the flight system being contained in the analog portion of the computer. This method will give very fast integration times but raises particular computational problems because of the limited accuracy and amplitude range of the analog variables.

The Indirect Method [3] for obtaining optimal control solutions converts the problem into a two-point boundary-value problem. A solution can then be found by iteratively searching the initial condition space for the unknown initial condition vector of the co-state variables. This solution vector is more easily handled by a hybrid computer than the complete state and control vectors, which have to be stored when using Direct Methods [3].

The solution of the two-point boundary-value problem has been studied on an iterative analog computer [4-6]. The present research was initiated with the aim of implementing those techniques on a full hybrid computer and of studying some of the outstanding computational problems.

The structure of this report is as follows. The Indirect Method of solution is formulated in the next section. The following section describes the automation of feedback switching [4] which is necessary to ensure convergence of the iteration from an arbitrary initial guess; disturbances from the nominal solution, which can cause the analog variables to saturate, can also be handled. Contour Modification was found to be helpful in the original study [5] but required impractical analysis to use it. A numerical approach is described in the section titled Contour Modification. A simulation of on-line updating [6], which uses the techniques described in the previous sections, is presented in the section, Progressive Updating. The conclusions are given in the final section.

FORMULATION

Let the dynamic system be represented by the set of state equations,

$$\dot{\underline{x}}(t) = \underline{f}[\underline{x}(t), \underline{u}(t)] \tag{1}$$

where $\underline{x}(t)$ is an n-vector of state variables, $x_i(t)$, $i = 1, \dots n$; $\underline{u}(t)$ is an r-vector of control variables, $u_j(t)$, $j = 1, \dots r$, and $\underline{f}$ is an n-vector of known functions, f_k, $k = 1, \dots n$.

The problem is to determine a control vector $\underline{u}^*(t)$ (superscript * indicates an optimum quantity minimizing J) in the interval $t_0 \leq t \leq t_f$ such that a performance index $J = J[\hat{\underline{x}}(t_f)]$ is minimized. J is considered to be of the form,

$$J = K[\underline{x}(t_f)] + \int_{t_0}^{t_f} L[\underline{x}(t), \underline{u}(t)]\, dt \tag{2}$$

so that an additional state variable must be defined by

$$\dot{x}_0(t) = L[\underline{x}(t), \underline{u}(t)] = f_0[\underline{x}(t), \underline{u}(t)] \tag{3}$$

giving $\hat{\underline{x}}(t)$ as an $(n + 1)$ vector of state variables, $x_i(t)$, $i = 0, 1, \dots n$.

The function $\underline{u}^*(t)$ can be determined from Pontryagin's Maximum Principle by forming the Hamiltonian,

$$\mathcal{H}[\underline{x}(t), \underline{p}(t), \underline{u}(t)] = \sum_{i=0}^{n} p_i(t)\ f_i[(\underline{x}(t), \underline{u}(t)] \tag{4}$$

where $\underline{p}(t)$ is an $(n+1)$ vector of co-state (adjoint) variables, $p_i(t)$, $i = 0, 1, \ldots n$ with $p_0(t) = -1$.

The function $\underline{p}(t)$ is chosen such that three necessary conditions are satisfied [7] as follows:

1. Canonical Equations:

$$\begin{aligned} \dot{x}_j &= \frac{\partial \mathcal{H}}{\partial p_j(t)} \quad ; \quad j = 1, \ldots n \quad . \\ \dot{p}_j &= -\frac{\partial \mathcal{H}}{\partial x_j(t)} \quad ; \quad j = 1, \ldots n \quad . \end{aligned} \tag{5}$$

2. Boundary Conditions:

Initial:

$$\underline{x}(t_0) = \underline{X}^0 \quad \text{(given - present state)} \tag{6}$$

final:

$$\underline{p}(t_f) = \frac{\partial K}{\partial \underline{x}(t)}\bigg|\ \underline{x}(t_f) \quad \text{(transversality condition)} \tag{7}$$

or

$$\underline{x}(t_f) = \underline{X}^f \quad \text{(given - target state)} \quad . \tag{8}$$

3. Maximization of the Hamiltonian:

$$\mathcal{H}[\underline{x}(t), \underline{p}(t), \underline{u}^*(t)] \geq \mathcal{H}[\underline{x}(t), \underline{p}(t), \underline{u}(t)] \tag{9}$$

for all t, $t_0 \leq t \leq t_f$.

Thus, $\underline{u}^*(t)$ can often be found by solving the equation,

$$\frac{\partial \mathcal{H}}{\partial \underline{u}(t)} = 0 \quad . \tag{10}$$

The Direct Method of solution proceeds by choosing a nominal $\underline{u}(t)$, integrating the state equations forward from t_0 with starting values $\underline{X}^0$, and storing $\underline{x}(t)$; then the co-state equations are integrated backward from t_f with starting values $\underline{p}(t_f)$ and using the stored $\underline{x}(t)$, such that necessary conditions 1 and 2 are satisfied; $\underline{u}(t)$ is adjusted iteratively until condition 3 is satisfied.

The Indirect Method of solution uses equation (10) to replace $\underline{u}(t)$ as a function of $\underline{x}(t)$ and $\underline{p}(t)$ in the canonical equations, which are then integrated forward with a nominal choice of $\underline{p}(t_0)$. The initial co-state vector is adjusted iteratively until the final conditions in condition 2 are satisfied. Thus the problem is converted into a classical two-point boundary-value problem.

In connection with the Contour Modification technique, to be discussed later, an augmented performance index Jm can be formed:

$$Jm = J + \left[\sum_{i=1}^{n} \alpha_i e_i(t_f) \right]^2$$

where $\underline{\alpha}$ is an n-vector of weighting coefficients α_i, $i = 1, \dots n$, and $\underline{e}(t_f)$ is an n-vector of boundary errors e_i, $i = 1, \dots n$ being the errors in satisfying the required final conditions in condition 2. Contour Modification is concerned with the optimum selection of the weighting coefficients $\underline{\alpha}$.

FEEDBACK SWITCHING

General Considerations

The stabilizing effect of feeding back a function of the boundary error $\underline{e}(t_f)$ on the iterative solution of trajectory optimization problems has been demonstrated [4-5], although the switching point and the magnitude of each change were determined manually. In order to consider on-line control updating, the solution must be obtained completely automatically, which implies computer control of feedback switching. A hybrid computer is ideal for this purpose in that the digital computer can easily generate the logical decision to reduce or increase the feedback according to the progress of the iteration.

The implementation of automatic feedback switching raises three basic questions:

1. What should be the initial level of feedback?

2. What fractional change in feedback should occur at each stage?

3. What criterion should determine the switching instant?

There is no simple answer to these questions and the various parameters must be determined from experience and by trial and error. This is a feature of any iterative method to some extent; for example, in gradient methods, step-size selection is always a difficult problem and stopping conditions are hard to define exactly. Program control of the iterative loop parameters will often be sufficient to ensure a convergent iteration.

In the preceding item 1, if the nominal choice results in the overload of some analog variable, successive increases in feedback can be made until stable state and co-state trajectories are achieved. Items 2 and 3 are strongly interactive. The fractional change can be under program control so that a larger number of stages can be used when a particular problem shows itself to be very sensitive, or vice versa. However, if switching occurs too soon, then instability in the subsequent stage is likely also.

With the object of trying to keep the overall iteration time as small as possible, a workable strategy is to maintain the chosen level of fractional change constant while trying to generate a convergent sequence of iteration stages by controlling the switching point. For example, if switching from one stage to the next occurs too soon, resulting in analog variable overload, then the feedback is switched back to the previous level and closer convergence attempted at that stage until subsequent switching results in a stable trajectory at the lower level of feedback. If this method breaks down, then a reduced fractional feedback change at each level is allowed.

By operating this two-level form of control, it should be possible to achieve a sequence of iteration stages which converges to the optimum solution with no feedback. The tradeoff for the two control levels is between increasing the time spent at each iteration stage by delaying the switching point and increasing the number of stages by reducing the fractional feedback change, both factors tending to increase the overall iteration time.

There is no precise criterion which can be used to determine the switching point at each stage. Each subproblem has an inherent suboptimal format because of the inclusion of boundary-error feedback, so that the necessary conditions for optimality, which can provide a stopping condition, are no longer valid. An empirical criterion, which has been used successfully in this work, is to set a lower bound on the gradient of the objective function taken over one hill-climbing step. If the change in objective function falls below this bound, then switching occurs and the feedback is reduced ready for the next iteration stage. Variation in the magnitude of this lower bound provides the control in switching level referred to above.

The Algorithm

The i-th stage of iteration can be expressed by the following relationships [4]:

1. Canonical Equations:

$$\begin{aligned} \dot{\underline{x}} &= \underline{f}(\underline{x},\underline{p}) + C_x^{(i)} \underline{w}_x(\underline{x},\underline{p}) \\ \dot{\underline{p}} &= \underline{g}(\underline{x},\underline{p}) + C_p^{(i)} \underline{w}_p(\underline{x},\underline{p}) \end{aligned} \tag{11}$$

where C_x, C_p are diagonal coefficient matrices; $\underline{w}_x(\underline{x}, \underline{p})$ is an n-vector chosen such that $\underline{e}_x(t_f) \to 0$; $\underline{w}_p(\underline{x}, \underline{p})$ is an n-vector chosen such that $\underline{e}_p(t_f) \to 0$; $\underline{e}_x(t_f)$ is the error in satisfying boundary conditions on $\underline{x}(t_f)$, and $\underline{e}_p(t_f)$ is the error in satisfying boundary conditions on $\underline{p}(t_f)$.

2. Algorithm for Computing $\underline{p}^{*(i)}(t_0)$: The term $\underline{p}^{*(i)}(t_0)$ is the value of the co-state initial vector which minimizes $J[\underline{\hat{x}}(t_f)]$ at the i-th stage,

$$\underline{p}^{(i|j+1)}(t_0) = \underline{p}^{(i|j)}(t_0) + \Delta p^{(i|j)} \tag{12}$$

where $\Delta p^{(i|j)}$ is the incremental change in the j-th approximation to $\underline{p}^{*(i)}(t_0)$, here given by $k\, \partial J^{(i|j)} / \partial \underline{p}^{(i|j)}(t_0)$ (steepest descent); $\Delta p^{(i|j)}$ produces a change in J of $\Delta J^{(i|j)}$.

3. Feedback Switching:

a. Decrease in feedback: if

$$\frac{\Delta J^{(i|j)}}{\Delta p^{(i|j)}} \leq \epsilon \tag{13}$$

where ϵ is a preset lower bound, then

$$C^{(i+1)} = \mu^{(i)} C^{(i)} \tag{14}$$

where $\mu^{(i)} < 1$ determines the reduction in contribution of $\underline{w}(\underline{x}, \underline{p})$ for the (i + 1) th stage.

b. Increase in feedback: if

$$J[\underline{\hat{x}}(t_f)] \geq J_{max} \tag{15}$$

then

$$C^{(i+1)} = \frac{1}{\mu^{(i)}} C^{(i)} \tag{16}$$

where J_{max} is an upper bound imposed by saturation of the analog variables.

Hybrid Implementation

Program Structure. The basic structure of the hybrid program, written to implement the algorithm described in the previous section, is shown in Figure 1. The various segments are described as follows.

Main program EXEC is used to set up the hybrid interface, initialize the problem, and monitor the progress of the overall iteration. Subroutine FN controls the objective function evaluation and makes logical decisions as to feedback changing, gradient evaluation, parameter updating, and convergence. Subroutine INT controls the mode of the analog computer to integrate the canonical equations from given initial conditions and returns a value of objective function. Subroutine GRPRT calculates first and second derivatives of the objective function with respect to the parameters (co-state initial conditions). Subroutine SD controls the steepest descent updating of $\underline{p}(t_0)$ and the step size used in equation (12). Subroutine FDBK controls the changes in feedback and the level used.

A flow chart of the hybrid program is given in Figure 2; the parameters are listed as follows:

BE	Boundary error $\sum_{i=1}^{n} e_i(t_f)^2$
BE_{min}	Minimum BE, used as a stopping function for the overall iteration
FB	Current feedback level, $C^{(i)}$
VN	Current value of objective function
VS	Lowest value of VN stored
V_{max}	Upper bound on VN, indicating analog saturation
V_{min}	Lower bound on VN, used to accelerate convergence at each iteration stage

GRADV	Change in VN with respect to parameters recorded over one iterative step
ϵ	Lower bound on GRADV.

Computer Assignment. The canonical equations (11) are programmed on the analog computer (680). Integration of the equations is initiated by setting a sense line from the digital computer (640). The period of integration is controlled by means of a counter on the analog logic patch board; the period is under program control since the counter is loaded from the 640 by setting sense lines. The state and co-state variables, the objective function, and the boundary-error terms are all calculated on the 680 and transferred to the 640 through the ADC.

The state and co-state initial conditions, the feedback coefficients, (and the Contour Modification weighting coefficients when appropriate) are calculated on the 640 and transferred to the 680 through the DAC's and MDAC's. Various logical operations are made on the 680 to synchronize analog mode control with data transfer across the hybrid interface. All other operations are performed on the 640.

Application

The hybrid program, described in the previous section, has been applied to two second-order systems which were studied previously (see Appendix A). The same convergence problems as before were encountered at the final stage.

Adjustment trajectories for the system with terminal cost (System I) are shown in Figure 3a. These can be compared with those obtained using the manual-iterative analog version of the algorithm, shown in Figure 3b [5]. The need to improve the final convergence is clearly indicated in both cases. The method used in the previous study was Contour Modification, which is discussed in the next section.

Because the problem is much worse for more sensitive systems, such as System II, no further comparative results are available for application of the basic algorithm.

CONTOUR MODIFICATION

Introduction

The need for Contour Modification and the means for achieving it have been noted [5]. However, the necessity of using the system transition matrix, in solving for the required weighting coefficients $\underline{\alpha}$, makes the technique impractical for general use.

An attempt to develop a strictly numerical approach and its implementation on the hybrid computer are described in the following subsections. The idea behind this technique is that as the overall iteration progresses, much information about the topology of the objective function surface is generated through repeated gradient calculations. It would appear reasonable that, given the means for achieving Contour Modification, use could be made of this topological information in choosing the necessary weighting coefficients. The concept is therefore one of a hill-climber operating on a "hill" which is constantly evolving towards an optimum shape, namely hyperspherical.

Criterion Function E

The orthogonality and eccentricity minimization criteria that were used for analytic calculation of the optimum weighting coefficients [5] are equivalent to the single criterion that the augmented objective function shall have eigenvalues which are all equal in magnitude. This condition can be recognized by examining the matrix of second derivatives of the objective function. When the off-diagonal terms are all zero and the diagonal terms all equal, the objective function will be hyperspherical. Thus the optimum weighting coefficients can be determined from the minimization of a criterion function based on the second derivative matrix.

Let

$$H = [h_{ij}] \quad ; \quad i,j = 1, \dots n$$

be the matrix of second derivatives of the augmented objective function Jm with respect to $\underline{p}(t_0)$. Then a suitable criterion function is given by

$$E = E_1 + E_2 \tag{17}$$

where

$$E_1 = \sum_{i=1}^{n-1} (h_{ii} - h_{jj})^2 \quad ; \quad j = i + 1 \tag{18}$$

$$E_2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{1}{2} h_{ij} h_{ji} \quad ; \quad i \neq j \quad . \tag{19}$$

Generation of E should be well suited to the hybrid computer in that the first and second derivatives can be determined by a finite difference method [8] involving repreated solution of the state and co-state equations. This method would seem preferable to that of using influence coefficients [9] to determine the gradients, with the attendant large increase in the number of equations to be programmed.

Unfortunately, E itself is extremely sensitive to the values of $\underline{\alpha}$, with the result that implementation on the hybrid computer has not been very successful. However, experiments with all digital computations to obtain $\underline{\alpha}^*$ through minimization of E have been performed successfully.

Minimization of E

The choice of algorithm for minimizing E is dictated primarily by the way in which E is calculated. When H and hence E are determined by a finite-difference method, no analytic expression is available for computing the gradient of E with respect to $\underline{\alpha}$. Grad E would itself have to be determined by a finite-difference method, leading to a very rapid increase in the number of function evaluations through solution of the system of canonical equations.

One computation of E takes $(2n^2 + 1)$ equation solutions [8]. One computation of Grad E takes 2n evaluations of E, thus $2n(2n^2 + 1)$ equation solutions.

If influence coefficients are used to determine the gradients of the objective function, then an analytic expression for Grad E can be determined (see Appendix B). The number of function evaluations is reduced substantially, at the expense of $2n^2$ additional equations to program.

It is preferable to use a direct-search type of algorithm, which does not require gradients, for minimizing E. The sensitivity of E, mentioned above, poses a problem for this type of search. However, various sophisticated algorithms have been developed, among them Zangwill's modification [10] of Powell's method [11], generally considered to be one of the most effective techniques [12]. This type of search requires repeated linear minimizations which can be performed in a variety of ways, among them cubic interpolation, quadratic approximation, and Golden Section search [12]. The latter method was used for the hybrid formulation because of its basic simplicity; although requiring more function evaluations, it is less sensitive to errors in data transfer across the hybrid interface.

Implementation

Digital Computation. Various algorithms have been applied to the problem of finding $\underline{\alpha}^*$ for System I. The results are presented in Table 1. The label "formula" refers to E being evaluated through the solution of the state transition matrix, thereby allowing gradient calculation in a closed form. "Analytic" refers to a closed-form solution of the optimization problem for $\underline{\alpha}$ — possible for this simple case. In all other iterations, E was obtained through a finite-difference evaluation of H.

Although Zangwill's algorithm used with GOLD1 required more than twice as many function evaluations as with QUADF, the simplicity of the method is generally more reliable, as evidenced by the results shown for System II. Although only second order, this system displays an objective function V and a criterion function E which are extremely sensitive to their respective parameters.

Hybrid Computation. The results of attempting a hybrid minimization of E are presented in Table 2. Convergence is erratic, the principal source of trouble being the evaluation of E. Computing H via finite differences, and then E from the elements of H, involves several subtractions of nominally similar quantities. The accuracy of the A-D and D-A converters in the hybrid interface is such that these subtractions can introduce large errors into the iteration.

It was hoped to combine the iteration on $\underline{\alpha}$ with the feedback iteration on $\underline{p}(t_0)$, described earlier. This would have allowed an evaluation of the "adaptive hill" concept discussed in the introduction to this section. However, the memory capacity of the EAI 640 computer is too small for the two programs to be loaded together so the double iteration could not be attempted.

TABLE 1. COMPUTATION OF $\underline{\alpha}^*$ (DIGITAL)

Algorithm	Iterations	Function Evaluations	Y_{min}	α_1^*	α_2^*
System I					
Z/QUADF	4	44	0.0219	2.239	-1.646
Z/GOLD1	3	94	0.01925	2.223	-1.632
CONJ/GOLD1 (formula)	11	217	0.00787	2.225	-1.634
CONJ/CUBIC (formula)	4	38	0.00028	2.232	-1.641
Z/GOLD1 (formula)	3	94	0.01926	2.223	-1.632
Z/QUADF (formula)	4	40	0.0998	2.229	-1.634
HYBRID	3	111	1.2500	2.017	-1.463
ANALYTIC				2.230	-1.640
System II					
Z/GOLD1	10	548	0.000	19.39	-2.209
Z/QUADF		No Convergence			
ANALYTIC				19.90	-2.275

Key: Z — Zangwill's Direct Search
CONJ — Conjugate Gradient Search
QUADF — Linear Search with Quadratic Fit
CUBIC — Linear Search with Cubic Fit
GOLD1 — Linear Search with Golden Sections

TABLE 2. COMPUTATION OF $\underline{\alpha}^*$ (HYBRID)

Criterion Function (Y)	Iterations	Function Evaluations	Y_{min}	α_1^*	α_2^*
E	6	188	0.5820	2.168	-1.628
$E^{1/2}$	3	111	1.2500	2.017	-1.463
$E^{1/2}$	3	153	6.442	2.362	-1.625

Nevertheless, the basic algorithm was used with optimum values of $\underline{\alpha}$ computed in the all-digital program. Adjustment trajectories for System I are shown in Figure 4. Comparison with Figure 3a shows the effectiveness of Contour Modification in improving the overall convergence.

A hand-plot of the progress of the iteration for System II is given in Figure 5; an automatic plot of the same adjustment trajectory proved to be too crowded to be meaningful.

Discussion

The results presented in the last subsection indicate that a criterion function for choosing the optimum weighting coefficients, to achieve Contour Modification, can be minimized through a direct-search type of algorithm. However, the features of the criterion function and its evaluation on the hybrid computer are such that inconclusive results have been obtained as to its usefulness. It is not clear at present what alternative criterion could be used that might lend itself better to minimization on the hybrid computer.

PROGRESSIVE UPDATING

Introduction

The techniques described in the sections on Feedback Switching and Contour Modification have been developed to enable solutions to trajectory optimization problems to be obtained on a hybrid computer. Optimal guidance is essentially a trajectory optimization problem so that nominal guidance laws can be computed. However, as noted previously, optimal solutions are generally very sensitive and large errors can appear toward the end of a trajectory interval, as a result of the limited accuracy obtainable from a hybrid computation.

A Progressive Computation or Progressive Updating method has been developed to overcome the error buildup [6]. Not only is the nominal guidance law corrected as the flight progresses but also the features for on-line guidance are contained in the method. Since the trajectory optimization problem is repeatedly solved for diminishing intervals, changes in the required end conditions or disturbances to the trajectory are easily accounted for. A description of the method follows and experimental results are presented.

Method

Solution of the trajectory optimization problem has been obtained in terms of the initial condition vector for the co-state variables. To reconstruct the optimal control function in real time, a simulation of the system co-state is required, to be used in conjunction with the measured (or estimated) system state vector. The interconnection of a high-speed computer model and real-time hardware (or plant) was suggested by Paiewonsky [13] and is shown in Figure 6.

The initial high-speed computation is carried out over the complete control interval $(t_f - t_0)$ and $\underline{p}^*(t_0)$ is transferred to the real-time co-state simulation. After a given time interval, τ, has elapsed, the plant state and co-state are sampled and transferred to the model. A new high-speed computation is then started over the reduced control interval $t_f - (t_0 + \tau)$ with $\underline{x}(t_0 + \tau)$ as the initial state vector and $\underline{p}(t_0 + \tau)$ as the first guess at the required optimal vector $\underline{p}^*(t_0 + \tau)$. When obtained, the new optimal initial co-state vector for the control interval $t_f - (t_0 + \tau)$ is transferred to the real-time co-state integrators and the plant trajectory proceeds from $\underline{x}(t_0 + \tau)$ and $\underline{p}^*(t_0 + \tau)$. The procedure is repeated at regular intervals τ throughout the overall control interval $(t_f - t_0)$ and thus the control is continually updated, ensuring a nearly optimum trajectory over the remaining time.

The time taken to perform the high-speed computation has been neglected in the preceding description but can be allowed for by using the model to predict the state of the plant after the computation time, $\Delta\tau$. In this way, for the r-th computation, $\underline{p}^*(t_0 + r\tau + \Delta\tau)$ is computed on the model instead of $\underline{p}^*(t_0 + r\tau)$, relying on the fact that $\underline{x}(t_0 + r\tau + \Delta\tau)$, as measured on the model, should be a close approximation to $\underline{x}(t_0 + r\tau + \Delta\tau)$ on the plant.

Hybrid Simulation

The updating scheme described above was simulated on the hybrid computer, although the allowance for computing time was not included. This omission greatly simplified the timing and synchronization of the simulation, which is described by the following steps:

1. From the given initial state $\underline{x}(t_0)$ and the guessed initial co-state $\underline{p}(t_0)$, the approximation to the optimal initial co-state $\underline{p}^{*(m)}(t_0)$ is computed on the model. Predetermined weighting coefficients $\underline{\alpha}(t_0)$ are used.

2. The plant equations are integrated over an interval from initial conditions $\underline{x}(t_0)$, $\underline{p}^{*(m)}(t_0)$; $\underline{x}(t_0 + \tau)$, $\underline{p}(t_0 + \tau)$ and the cost incurred $x_0(t_0 + \tau)$ are measured from the plant.

3. From the measured initial state $\underline{x}(t_0 + \tau)$ and the measured initial co-state $\underline{p}(t_0 + \tau)$, the approximation to the optimal initial co-state $\underline{p}^{*(m)}(t_0 + \tau)$ is computed on the model. Predetermined weighting coefficients $\underline{\alpha}(t_0 + \tau)$ are used.

4. Replacing t_0 by t_r, $r = 1, 2, \ldots, (q - 1)$, where

$$t_r = t_{r-1} + \tau$$

$$q = \frac{(t_f - t_0)}{\tau} \quad ,$$

steps 2 through 4 are repeated until $r = q - 1$.

5. With $r = q - 1$ and hence $t_{r+1} = t_f$, the final state $\underline{x}(t_f)$, final co-state $p(t_f)$, and final cost incurred $x_0(t_f)$ are measured from the plant.

6. The total cost incurred by the plant over the complete control interval, i.e., the performance index J, is given by

$$J[\hat{\underline{x}}(t_f)] = K[\underline{x}(t_f)] + \sum_{r=1}^{q} x_0(t_r) \qquad (20)$$

where

$$\hat{\underline{x}} = \begin{bmatrix} x_0 \\ \underline{x} \end{bmatrix} .$$

Both model and plant were simulated on the analog portion of the hybrid computer. Time scales were under digital control, with the model being run one thousand times faster than the plant.

Because of the difficulty of implementing the computation of $\underline{\alpha}$ on the hybrid computer, the weighting coefficients were computed for the various time intervals in an all-digital program and input as data to the hybrid program.

Experimental Results

To assess the effectiveness of progressive updating, the method should be applied to a sensitive system, in other words, one in which the trajectory computed over the complete control interval from $\underline{p}^{*(m)}(t_0)$ diverges from the true optimum trajectory.

Comparative trajectories for System I are shown in Figure 7. Clearly, the single trajectories computed from $\underline{p}^{*(m)}(t_0)$ (as obtained in Figure 4) are close to the optimum trajectories. (The latter were obtained from an all-digital solution as in Reference 6). Therefore, there is nothing to be gained from using Progressive Updating with this system.

However, this is not the case for System II. Comparative trajectories are shown in Figure 8 and divergence from the optimum is evident. The third trajectories in Figure 8 were obtained through application of the Progressive Updating scheme; strip-chart recordings of the same trajectories are shown in Figure 9. The effect of Progressive Updating in driving the state and co-state trajectories towards the optimum is clearly seen.

Comparative values of the performance index for the trajectories of Figure 8 are

Single computation: $J = 15.08$

Progressive updating: $J = 4.03$

Digital (optimum) computation: $J = 3.04$

These values were obtained from equation (20). Table 3 gives the values of $\underline{\alpha}$ used at each updating stage.

TABLE 3. $\underline{\alpha}^*$ USED FOR EACH STAGE OF PROGRESSIVE UPDATING

Updating Stage	$\underline{\alpha}^T$
0	(19.380 -2.210)
1	(12.510 -1.232)
2	(7.485 -0.560)
3	(3.980 -0.147)
4	(1.682 -0.047)
5	(0.518 0.081)

The principal on which Progressive Updating is founded is that trajectories generated from a hybrid solution (inherently having some error) will be most nearly optimal during the early part of the trajectory. Therefore, the more often updating can occur, the more accurate will be the overall trajectory. The faster a solution can be obtained, the more often updating can occur. Consequently, on-board guidance updating requires as fast a computing method as possible, hence the consideration of a hybrid mode. The net result is a classic tradeoff between speed of computing and the accuracy of an individual solution.

CONCLUSIONS

On-line solutions of trajectory optimization problems, for use in real-time guidance updating schemes, require a high-speed iterative computation. One approach is to exploit the rapid integration capability of a hybrid computer. Certain computational difficulties arise, however, because of the limited precision and amplitude range on an analog computer. Possible methods for overcoming these difficulties have been studied in this report with varying degrees of success.

Solution of the optimization problem is obtained in terms of the initial co-state vector. Convergence from an arbitrary initial guess is uncertain, due largely to the inherent instability in the canonical equations, causing saturation of analog variables. A feedback technique has been developed to force the state and co-state variables toward satisfying given boundary conditions, thereby maintaining stability in the trajectories. Although basically empirical in its implementation, the technique has been successfully applied to simple dynamical systems, simulated on the hybrid computer.

Final convergence of the feedback technique is, however, subject to the difficulties mentioned above since feedback is removed over the last stage. The known final boundary conditions have been used to stabilize the final convergence by augmenting the objective function in the solution space. The criterion used for such objective function contour modification has been to diagonalize the matrix of second derivatives, generating eigenvalues of equal magnitude.

The appropriate weighting coefficients for modification have been successfully calculated with an all-digital program. A similar hybrid program, however, was not successful in generating a stable solution. The extreme sensitivity of the criterion function itself has been identified as contributing to this difficulty, together with inherent noise in the analog and linkage portions of the computer.

A simulation of progressive control function updating was performed, using weighting coefficients calculated digitally. Both a high-speed model and a simulated real-time plant were programmed on the analog computer. Experimental results have shown that improved trajectory tracking can be obtained by regularly updating the optimal solution. The initial solution, computed by the feedback technique with contour modification, provides an appropriate control function over the initial stages of the trajectory. The tendency for this primary solution to be in error as the trajectory progresses is corrected by the updating scheme.

The hybrid computer has been shown to be useful in generating on-line optimal guidance functions. However, the stability and speed of the primary solution remains an area for further study.

APPENDIX A

SYSTEM EQUATIONS

The canonical equations, boundary conditions, and performance indices for the systems on which the algorithms were tested are given below:

1. System with Terminal Cost:

$$J = x_1^2(t_f) + x_2^2(t_f) + \int_{t_0}^{t_f} 5\,u^2\,dt$$

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= u \end{aligned} \quad ; \quad \begin{aligned} \dot{p}_1 &= 0 \\ \dot{p}_2 &= -p_1 \end{aligned} \quad ; \quad u = -0.1\,p_2 \quad .$$

$$\underline{x}(t_0) = \begin{bmatrix} 1.0 \\ 0 \end{bmatrix} \quad ; \quad \underline{p}(t_f) = \begin{bmatrix} 2\,x_1(t_f) \\ 2\,x_2(t_f) \end{bmatrix} \quad .$$

2. Lag plus Integrator System:

$$J = \int_{t_0}^{t_f} \left(x_1^2 + 5\,u^2\right) dt$$

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -x_2 + u \end{aligned} \quad ; \quad \begin{aligned} \dot{p}_1 &= -2\,x_1 \\ \dot{p}_2 &= -p_1 + p_2 \end{aligned} \quad ; \quad u = -0.1\,p_2$$

$$\underline{x}(t_0) = \begin{bmatrix} 1.0 \\ 0 \end{bmatrix} \quad ; \quad \underline{p}(t_f) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad .$$

APPENDIX B

ANALYTIC DETERMINATION OF GRAD E

The components of $H = [h_{ij}]$ $i, j = 1, \ldots n$, can be evaluated from successive measurements of the first derivatives m_i, $i = 1, \ldots n$.

At a nominal point s, measure $m_r^{(s)}$, $r = 1, \ldots n$. At successive points q, measure $m_r^{(q)}$, $r = q, \ldots n$; $q = 1, \ldots n$ where points q are obtained from the nominal point s by perturbing $p_q(t_0)$ by an amount $\Delta p_q(t_0)$ and holding all other components of $\underline{p}(t_0)$ fixed. Then,

$$h_{ii} = \frac{m_i^{(i)} - m_i^{(s)}}{2\,\Delta p_i(t_0)} \quad ; \quad i = 1, \ldots n$$

$$h_{ij} = \frac{m_i^{(j)} - m_i^{(s)}}{2\,\Delta p_j(t_0)} \quad ; \quad i > j \quad ; \quad \begin{array}{l} i = 1, \ldots n \\ j = 1, \ldots n \end{array}$$

$$= h_{ji}$$

$$E = \sum_{i=1}^{n-1} (h_{ii} - h_{i+1,i+1})^2 + \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{1}{2} (h_{ij} h_{ji}) \quad .$$

The derivative of E with respect to $\underline{\alpha}$ is required:

$$\frac{\partial E}{\partial \alpha_k} = 2 \sum_{i=1}^{n-1} (h_{ii} - h_{i+1,i+1}) \left(\frac{\partial h_{ii}}{\partial \alpha_k} - \frac{\partial h_{i+1,i+1}}{\partial \alpha_k} \right)$$

$$+ \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{1}{2} \left(h_{ij} \frac{\partial h_{ji}}{\partial \alpha_k} + h_{ji} \frac{\partial h_{ij}}{\partial \alpha_k} \right) \quad . \qquad \text{(B-1)}$$

Now $\underline{m}$ can be expressed in terms of influence coefficients [9], evaluated from additional differential equations:

$$m_i^{(j)} = \sum_{r=1}^{n} \left[V_{x_r}^{(j)} \, y_{ri} + V_{p_r}^{(j)} \, z_{ri} \right]$$

where

$$y_{ri}(t) = \frac{\partial x_r(t)}{\partial p_i(t_0)}$$

$$z_{ri}(t) = \frac{\partial p_r(t)}{\partial p_i(t_0)}$$

$$V_{x_r}(t) = \frac{\partial V}{\partial x_r(t)}$$

$$V_{p_r} = \frac{\partial V}{\partial p_r(t)} \quad .$$

Writing

$$\Delta m_i^{(j)} = m_i^{(j)} - m_i^{(s)}$$

$$= \sum_{r=1}^{n} \left[\Delta V_{x_r}^{(j)} \, y_{ri} + \Delta V_{p_r}^{(j)} \, z_{ri} \right] \quad ,$$

then

$$\frac{\partial \Delta m_i^{(j)}}{\partial \alpha_k} = \sum_{r=1}^{n} \frac{\partial \left[\Delta V_{x_r}^{(j)}\right]}{\partial \alpha_k} y_{ri} + \frac{\partial \left[\Delta V_{p_r}^{(j)}\right]}{\partial \alpha_k} z_{ri}$$

$$= \frac{\partial h_{ij}}{\partial \alpha_k} \quad . \qquad (B-2)$$

Substituting equation (B-2) into equation (B-1) allows $\partial E/\partial \alpha_k$ to be evaluated, since $\Delta_{V_{\underline{x}}}$ and $\Delta V_{\underline{p}}$ are functions of $\underline{\alpha}$.

REFERENCES

1. Horn, H.J.; Martin, D.T.; and Chandler, D.C.: An Iterative Guidance Scheme and Its Application to Lunar Landing. NASA TN D-2869, Marshall Space Flight Center, July 1965.

2. Baker, C.D.; Causey, W.E.; and Ingram, H.L.: Mathematical Concepts and Historical Development of the MASCOT Guidance Technique for Space Vehicles. NASA TM X-64608, Marshall Space Flight Center, July 1971.

3. Kopp, R.E.; and McGill, R.: Several Trajectory Optimization Techniques. Section in Computing Methods in Optimization Problems, A.V. Balakrishnan and L.W. Neustadt, eds., Academic Press, Inc., New York, 1964.

4. Neely, P.L.; and Roberts, A.P.: An Analog Technique for Solving Trajectory Optimization Problems. Int. J. Control, Vol. 9, No. 6, 1969, p. 695.

5. Neely, P.L.; and Roberts, A.P.: An Improved Analog Technique for Solving Trajectory Optimization Problems. Int. J. Control, Vol. 13, No. 1, 1971, p. 33.

6. Neely, P.L.; and Roberts, A.P.: A Progressive Computation Technique for Updating Optimal Control Functions. Int. J. Control, Vol. 13, No. 2, 1971, p. 325.

7. Athans, M.; and Falb, P.L.: Optimal Control: An Introduction to the Theory and Its Applications. McGraw-Hill Book Co., New York, 1966.

8. Sharp, D.R.; and Enns, M.: A Hybrid Computer Method for Computing Optimal Controls. Proceedings of the AICA/IFIP Convention, Munich, Germany, 1970, p. 152.

9. Meissinger, H.F.: The Use of Parameter Influence Coefficients in Computer Analysis of Dynamic Systems. Proceedings of the Western Joint Computer Conference, May 1960.

10. Zangwill, W.I.: Minimizing a Function Without Calculating Derivatives. Computer J., Vol. 10, 1967-68, p. 293.

REFERENCES (Concluded)

11. Powell, M. J. D.: An Efficient Method for Finding the Minimum of a Function of Several Variables Without Calculating Derivatives. Computer J., Vol. 7, 1964, p. 155.

12. Beveridge, G. S. G.; and Schechter, R. S.: Optimization: Theory and Practice. McGraw-Hill Book Co., New York, 1970.

13. Paiewonsky, B.: Time Optimal Control of Linear Systems with Bounded Controls. Section in Proceedings of International Symposium on Non-Linear Differential Equations and Non-Linear Mechanics, J. P. Lasalle and S. Lefshetz, eds., Academic Press, Inc., New York, 1963, p. 333.

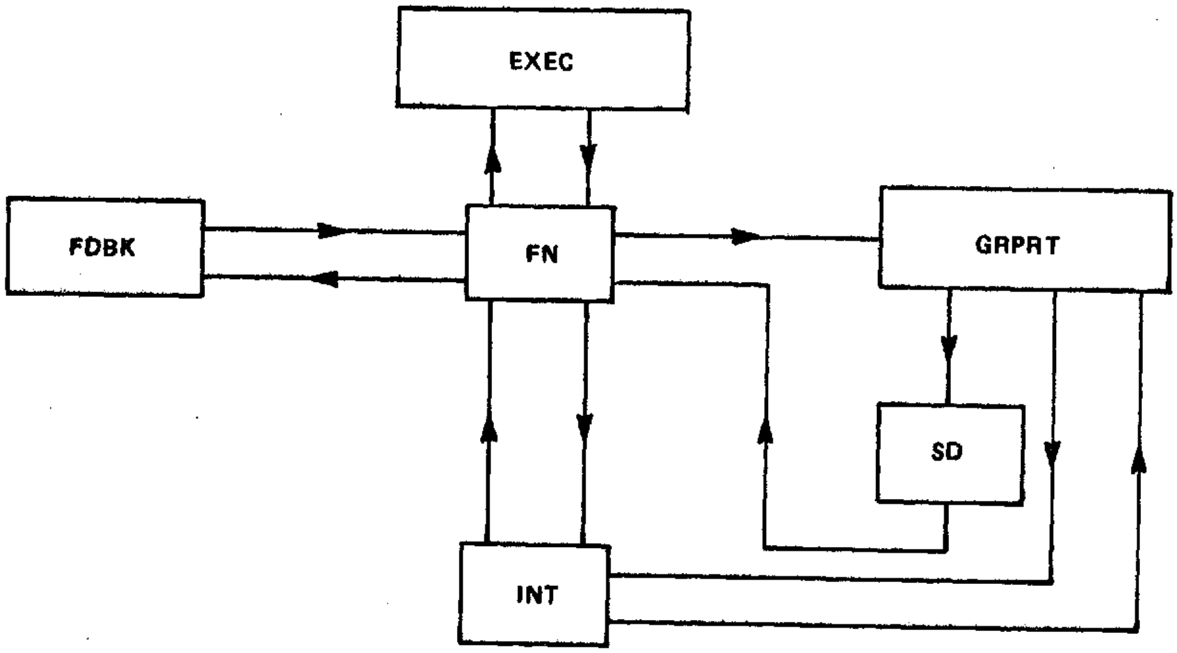


Figure 1. Hybrid program structure.

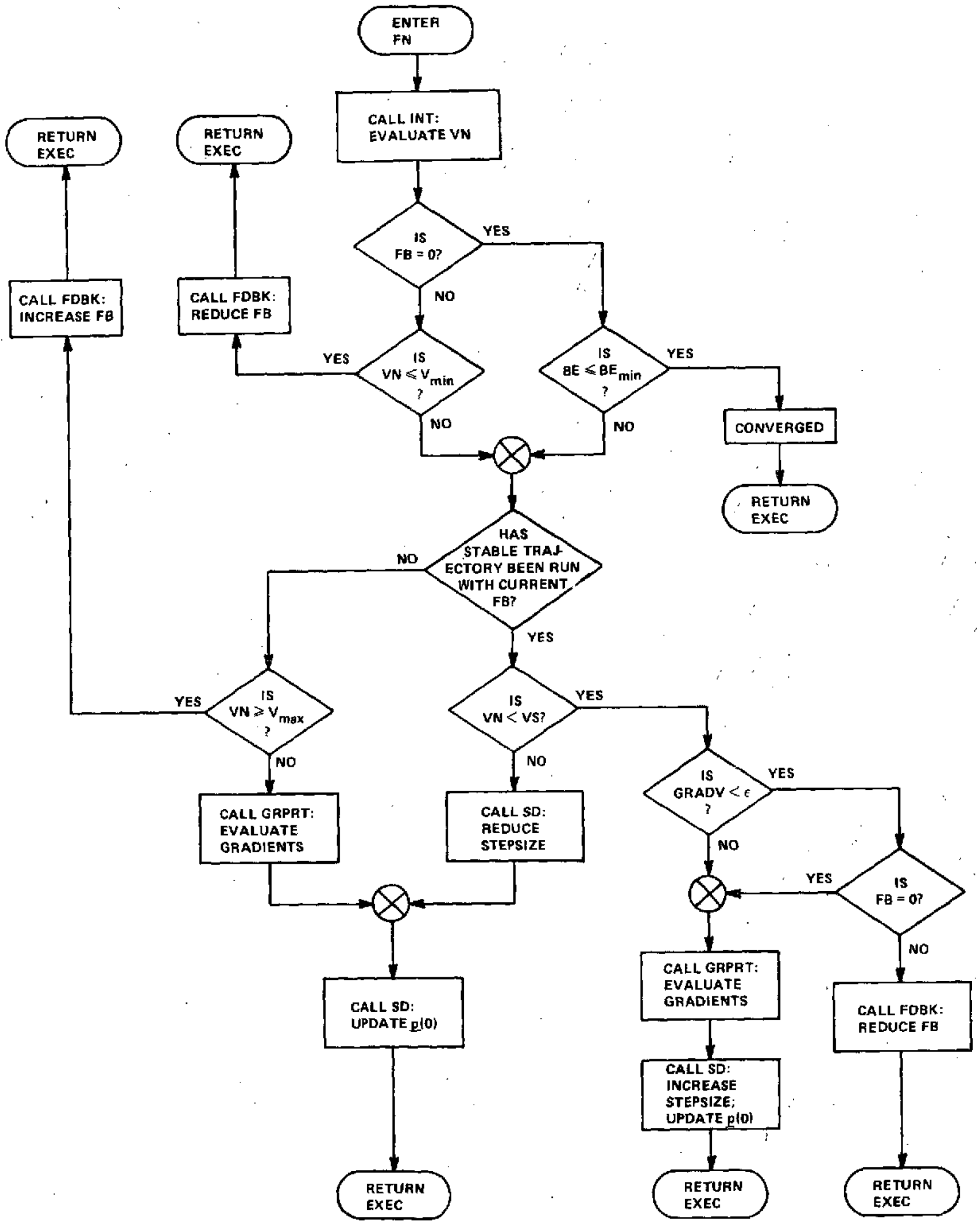
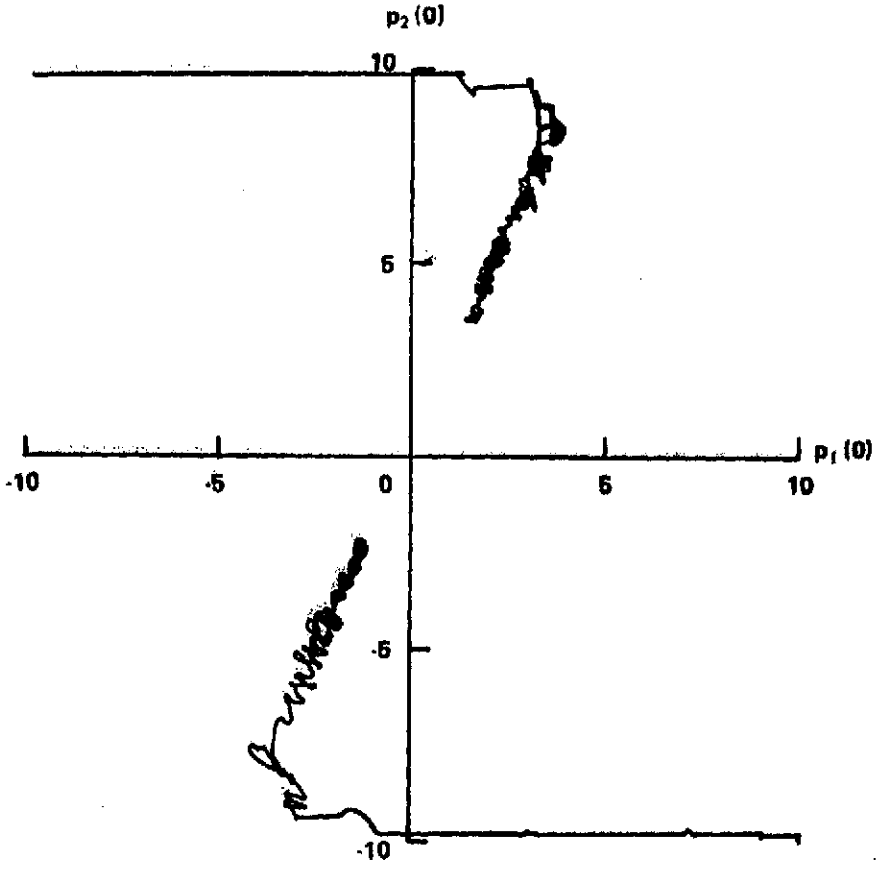
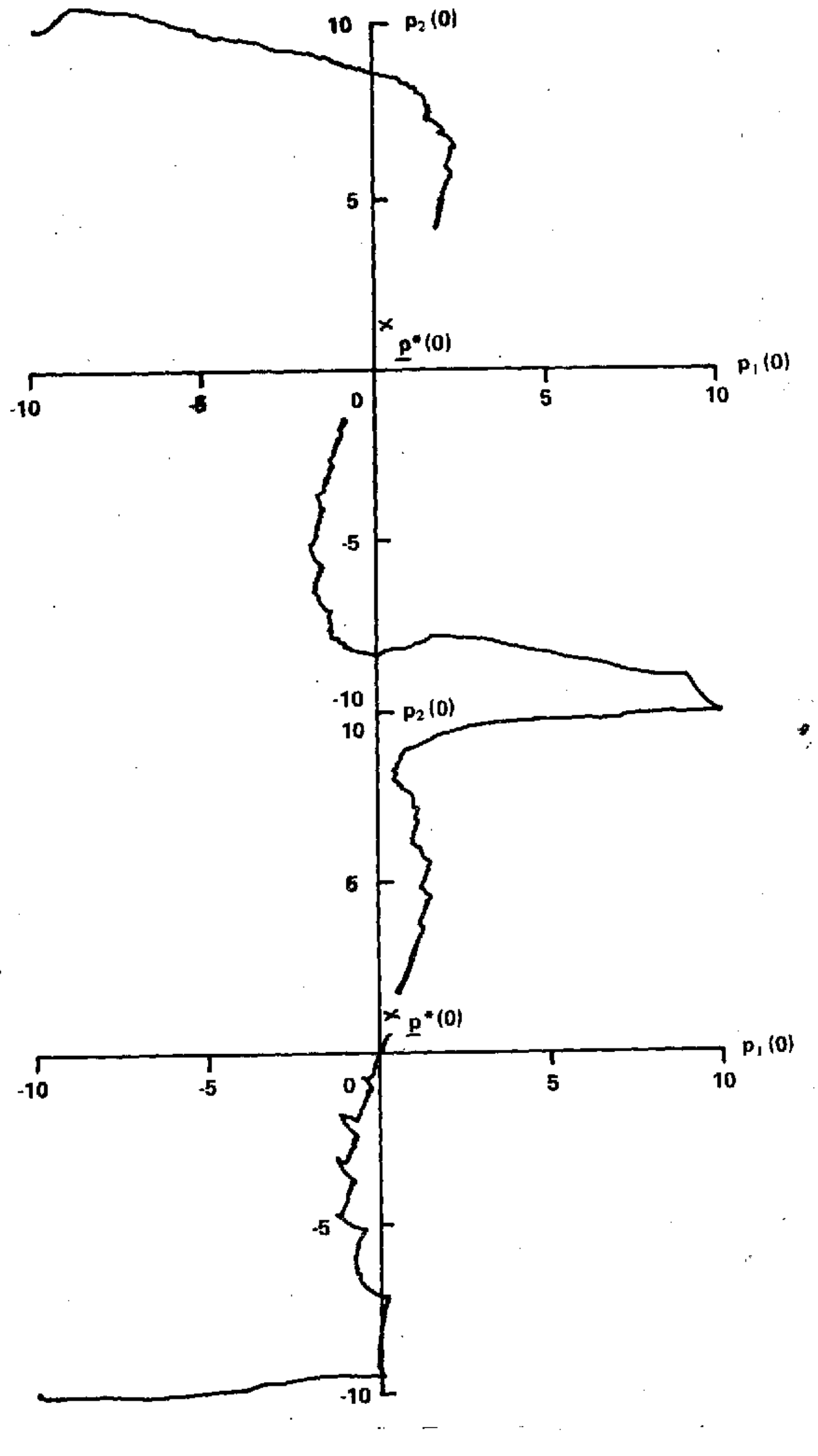


Figure 2. Flow chart of hybrid program.



a. Automatic trajectories.

Figure 3. Adjustment trajectories for System I.



b. Manual trajectories.

Figure 3. Adjustment trajectories for System I.

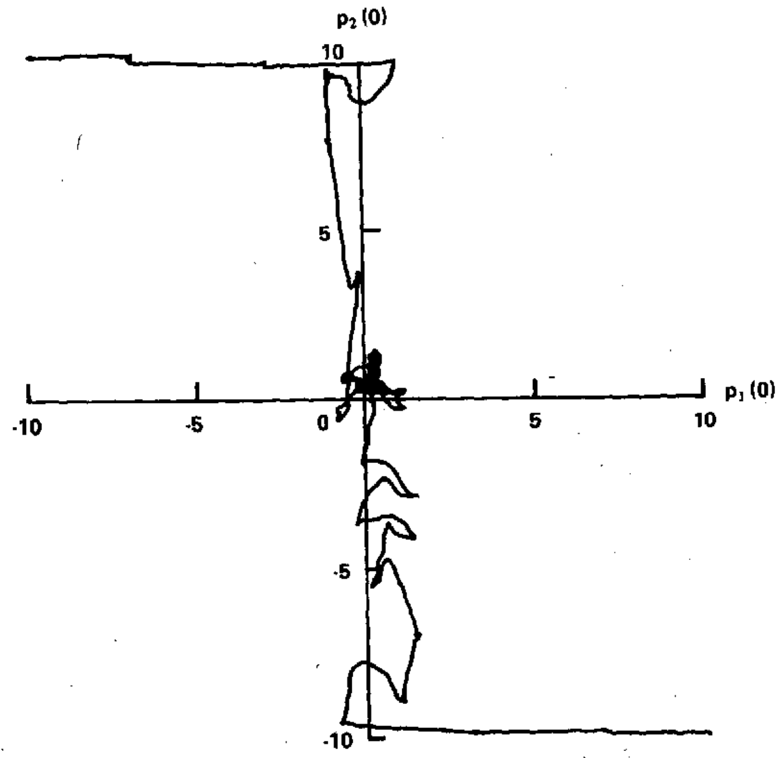


Figure 4. Adjustment trajectories for System I (Countour Modification).

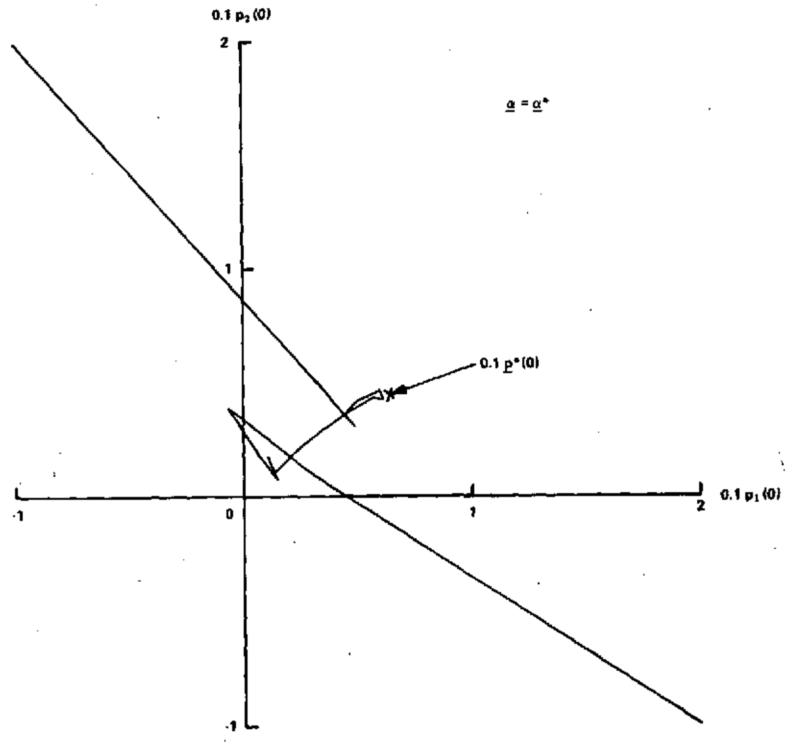


Figure 5. Adjustment trajectories for System II (hand plot).

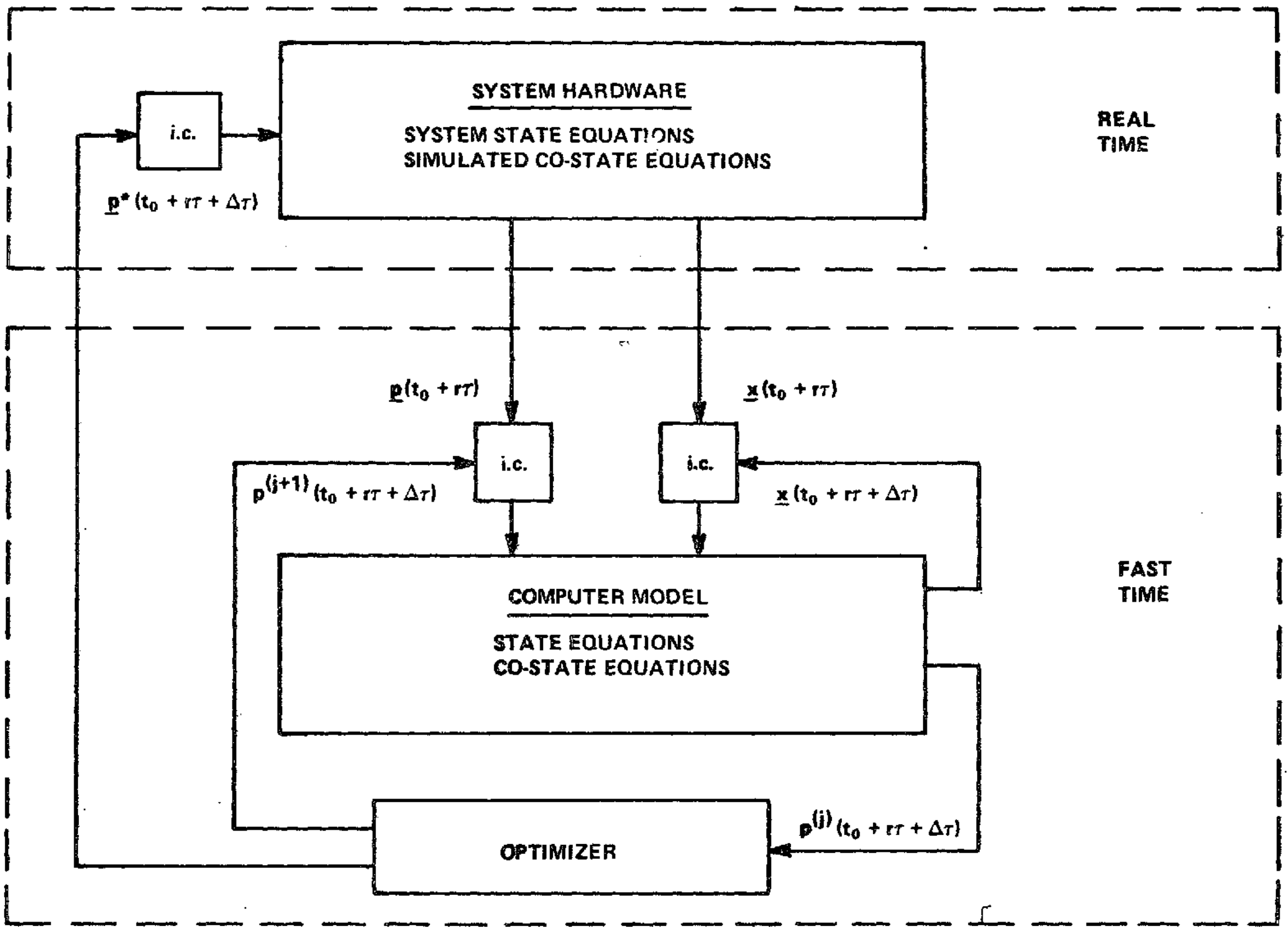
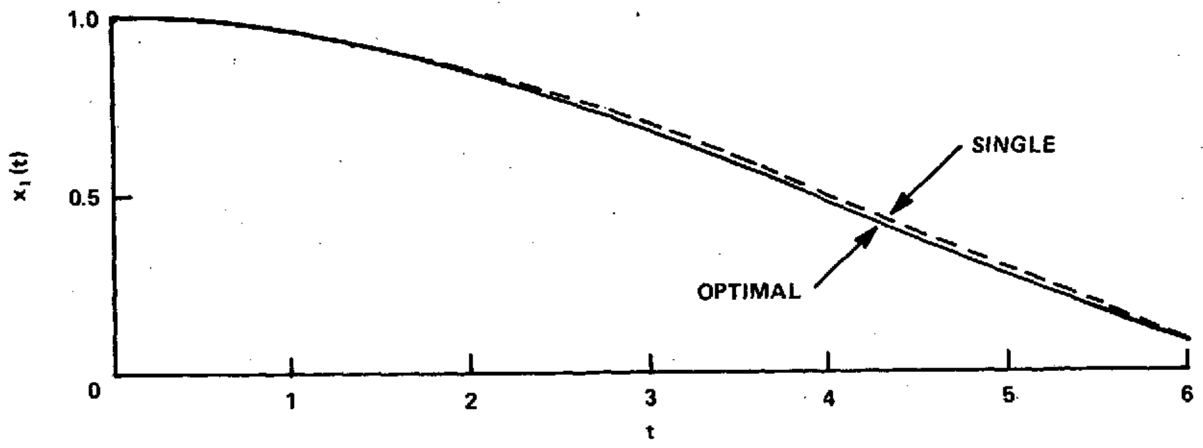
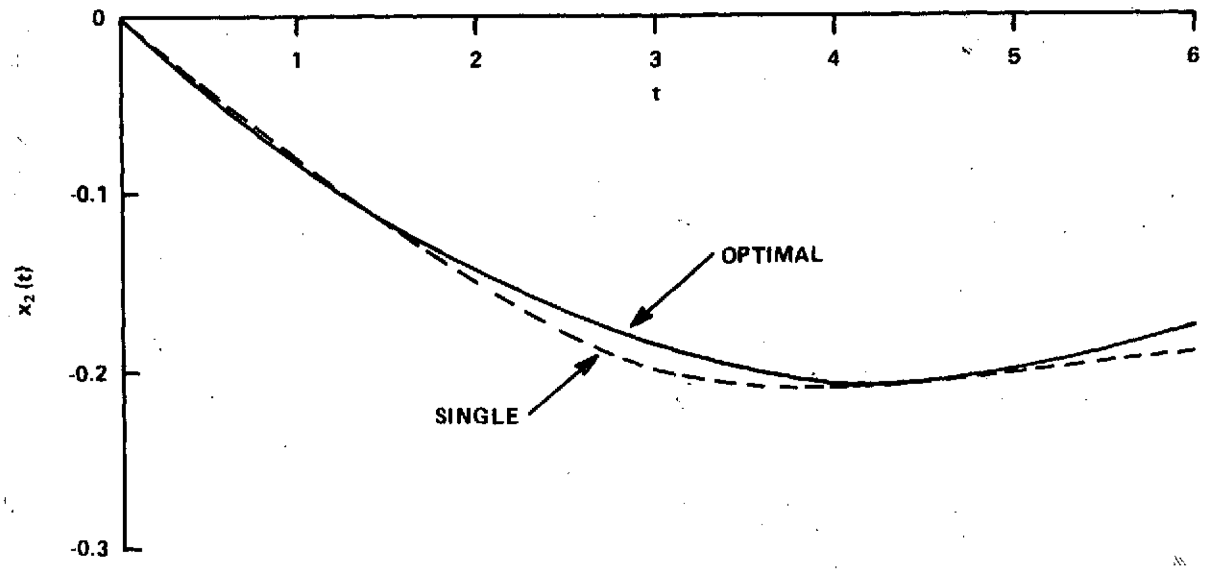

SYSTEM HARDWARE
SYSTEM STATE EQUATIONS
SIMULATED CO-STATE EQUATIONS
REAL TIME
i.c.
$\underline{p}^*(t_0 + r\tau + \Delta\tau)$
$\underline{p}(t_0 + r\tau)$
$\underline{x}(t_0 + r\tau)$
i.c.
i.c.
$p^{(j+1)}(t_0 + r\tau + \Delta\tau)$
$\underline{x}(t_0 + r\tau + \Delta\tau)$
COMPUTER MODEL
STATE EQUATIONS
CO-STATE EQUATIONS
FAST TIME
OPTIMIZER
$p^{(j)}(t_0 + r\tau + \Delta\tau)$

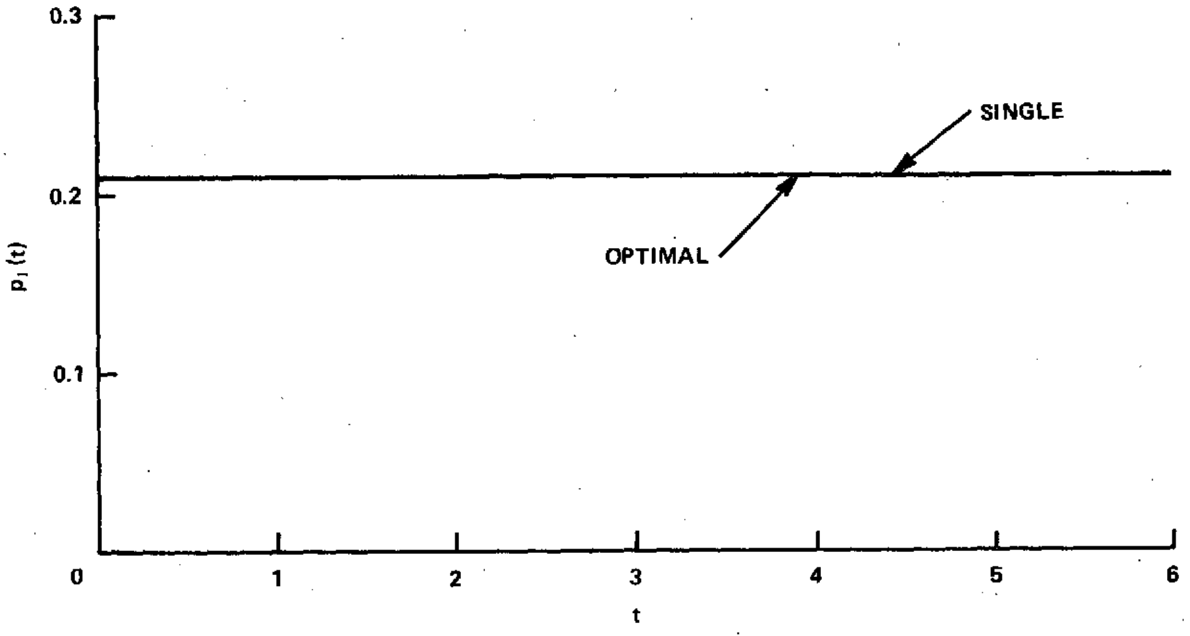
Figure 6. Progressive Updating model.

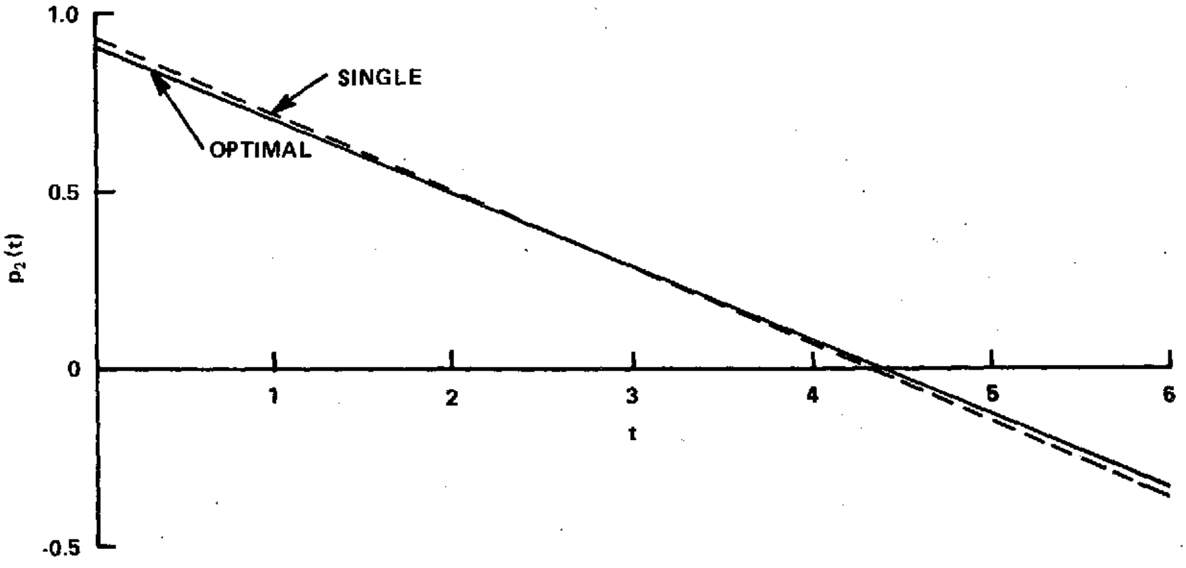




a. State trajectories.

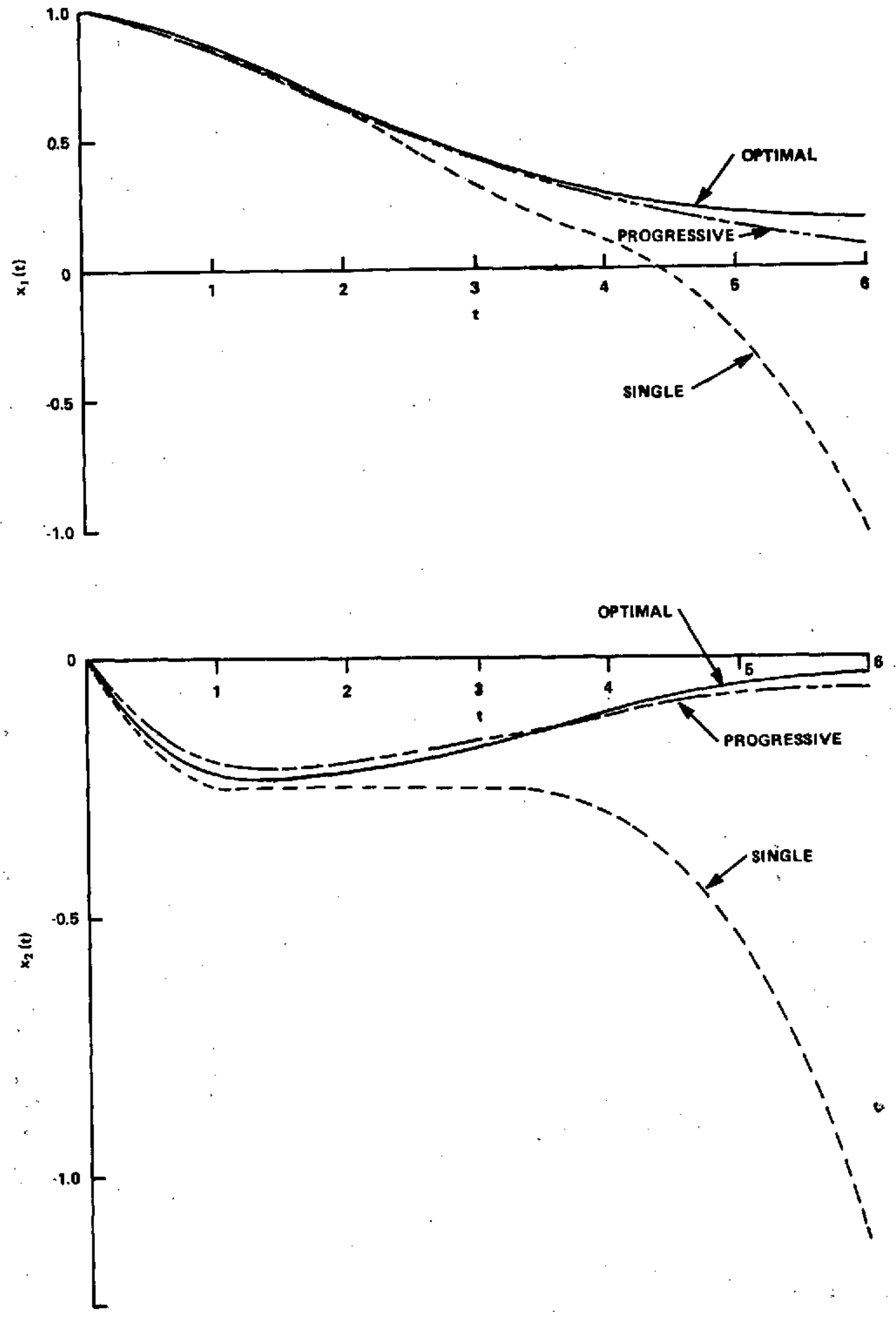
Figure 7. Comparative trajectories for System I.





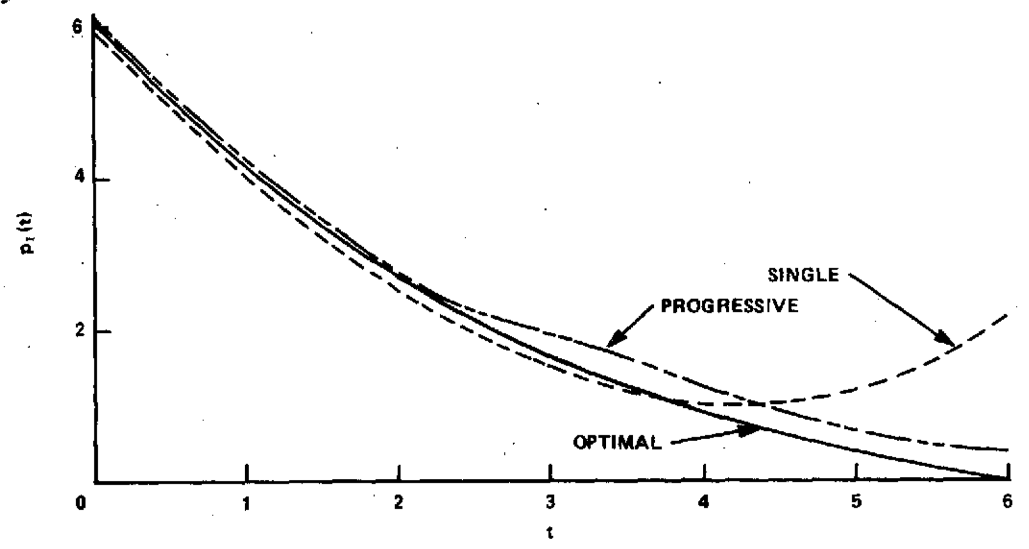
b. Co-state trajectories.

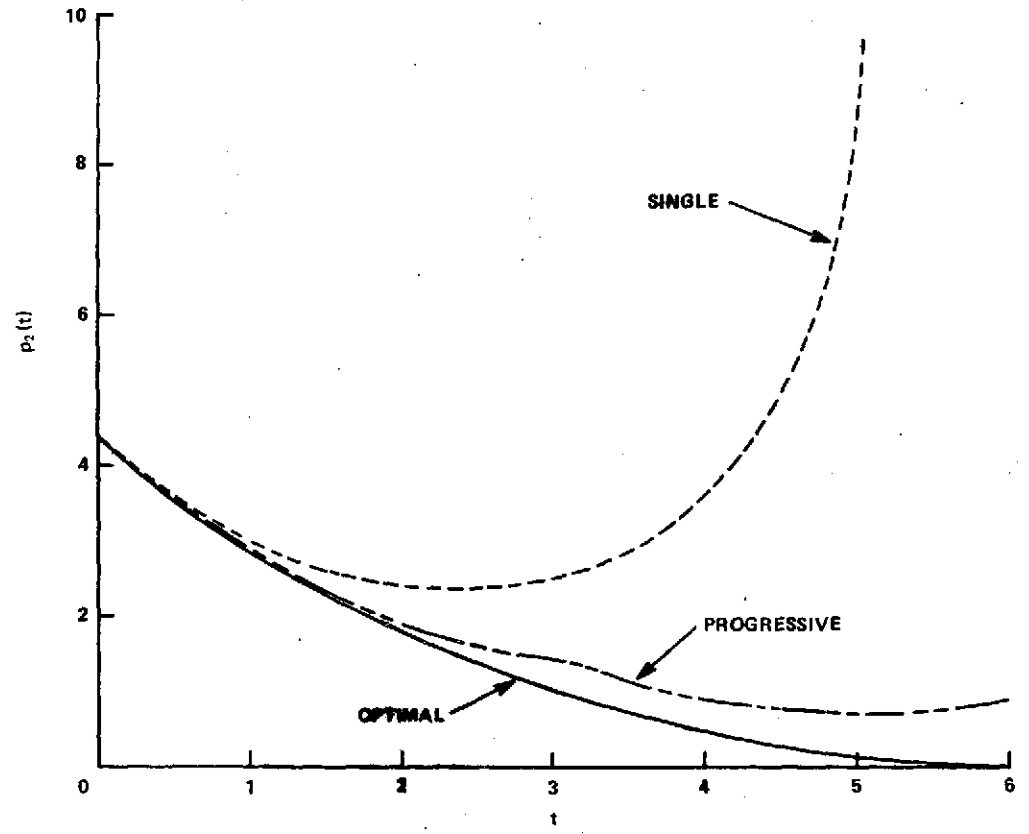
Figure 7. Comparative trajectories for System I.



a. State trajectories.

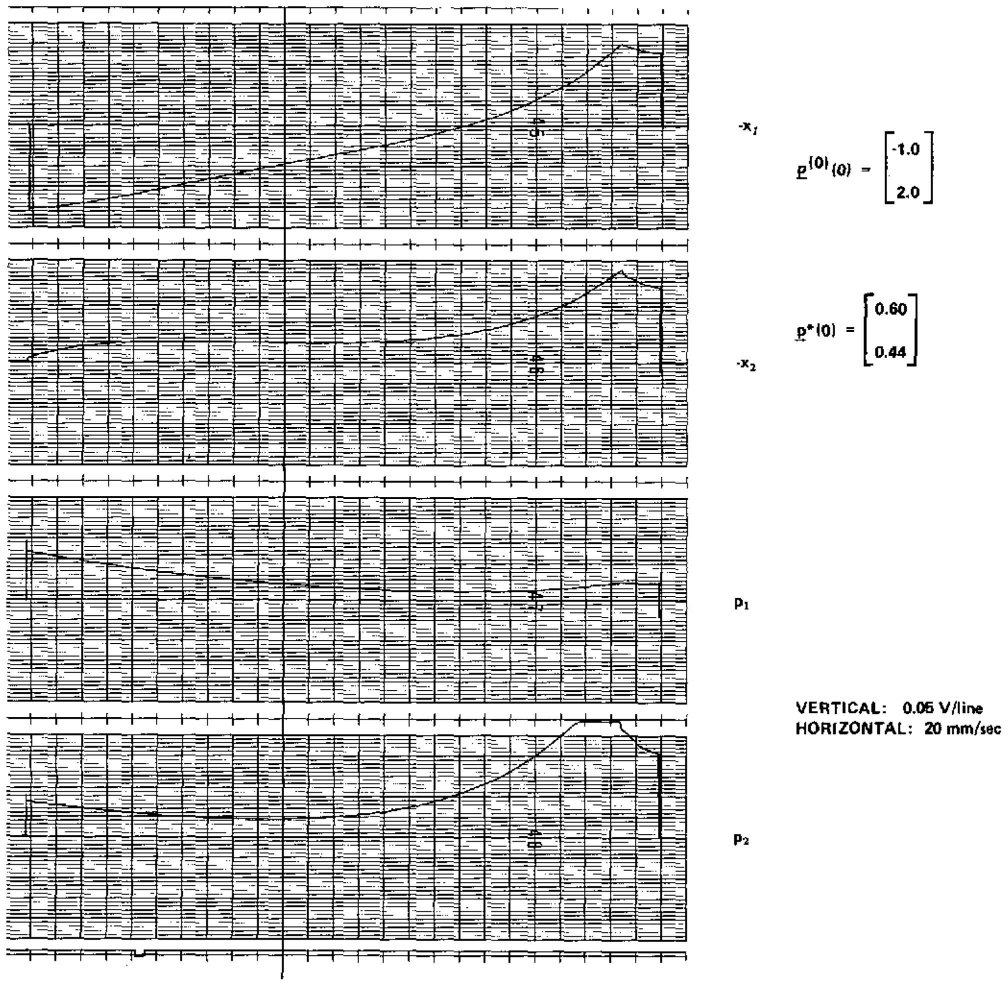
Figure 8. Comparative trajectories for System II.





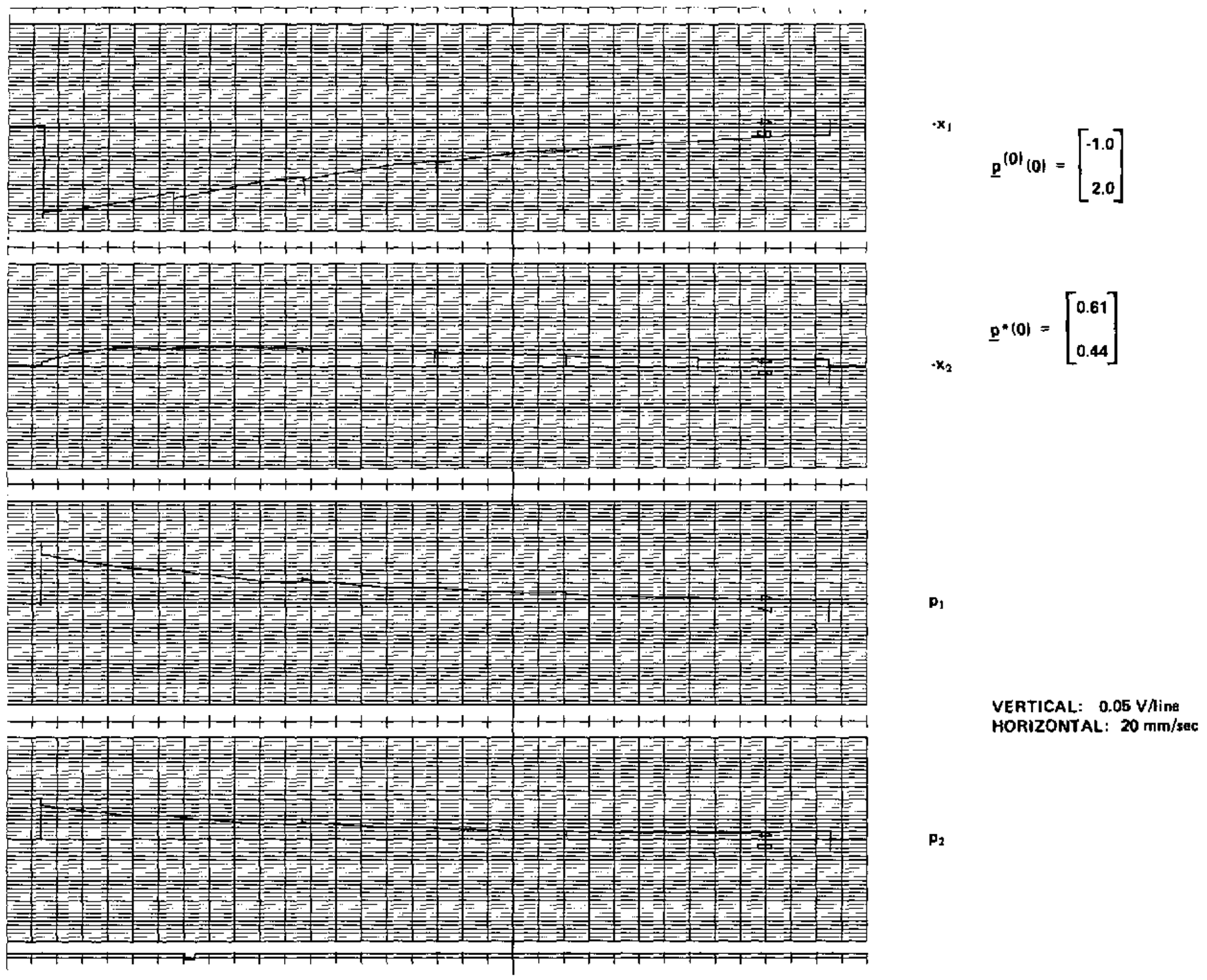
b. Co-state trajectories.

Figure 8. Comparative trajectories for System II.



a. Single trajectories.

Figure 9. Strip chart for System II.



b. Progressive trajectories.

Figure 9. Strip chart for System II.

APPROVAL

TECHNIQUES FOR TRAJECTORY OPTIMIZATION USING A HYBRID COMPUTER

By P. L. Neely

The information in this report has been reviewed for security classification. Review of any information concerning Department of Defense or Atomic Energy Commission programs has been made by the MSFC Security Classification Officer. This report, in its entirety, has been determined to be unclassified.

This document has also been reviewed and approved for technical accuracy.

[illegible]

J. A. LOVINGOOD
Director, Systems Dynamics Laboratory

☆ U.S. GOVERMENT PRINTING OFFICE 1975–640-451/217 REGION NO. 4

DISTRIBUTION

INTERNAL

DS30
Dr. Bucher

ED01
Dr. Lovingood

ED11
Dr. Blair

ED15
Dr. Winder

ED21
Mr. Ryan

ES11
Dr. Teuber

CC
Mr. L. D. Wofford, Jr.

AS61 (2)

AS61L (8)

AT01 (6)

EXTERNAL

Computer Sciences Corp.
8300 Whitesburg Dr.
Huntsville, Ala. 35802
Attn: Dr. P. L. Neely (6)

Scientific and Technical Information Facility (25)
P. O. Box 33
College Park, Md. 20740
Attn: NASA Representative (S-AK/RKT)